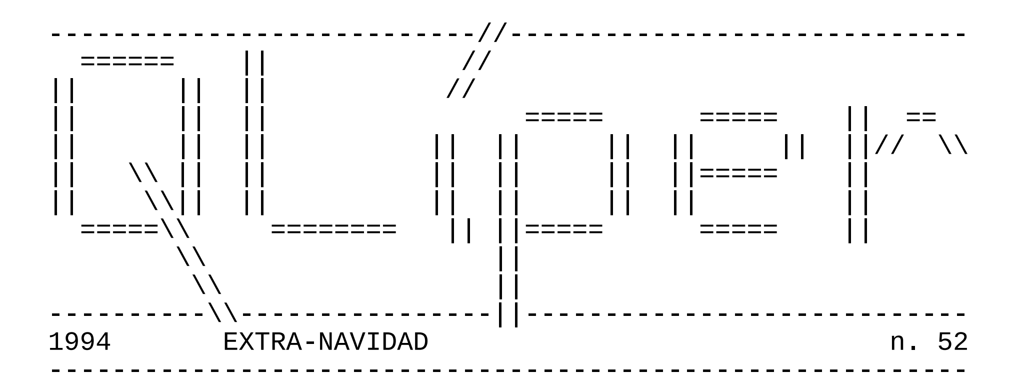


QLíper
Redactor: Salvador Merino
Tel. +34-(9)5-2475043
Cerámicas Mary
Ctra. Cádiz (Torreblanca del Sol)
ES-29640 FUENGIROLA

Cuota Socio

La cuota socio es anual (Comienza en Enero y termina en Diciembre).
La salida de QLíper es TRIMESTRAL (4 discos al año).

Cuota 1995 (4 discos):

ESPAÑA............ 1.000 Pesetas
EUROPA............ 7 - US $10 - 1.500 Pesetas
RESTO MUNDO ...... US $15 - 2.000 Pesetas

Editorial

Os recuerdo que con este disco finaliza el año 1994, y es la hora de comenzar a enviar la cuota de 1995 y alguna cosilla para llenar los discos.

Sobre la contabilidad del Club, he de decir que hay un superávit de alrededor de 19.000 Ptas acumulado de otros años. Aunque ha sido gracias a la austeridad de vuestro tesorero que ha reducido costes al máximo, porque a principios de año el superávit era de unas 5.000 Ptas, los ingresos por cuotas han sido de 21.500 Ptas, y los gastos minimos han sido de 14.640 Ptas, pero he ahi un milagro que no es tal, ya que esa diferencia de 10.000 Ptas ha salido de la copia de programas PD/ShareWare y atrasados. No obstante, gracias a ese superávit se ha conseguido reducir todavía más la cuota de socio.

Otra cosa que habréis notado nueva, es que estoy utilizando discos 3.5" HD formateados a 720K y boquete HD tapado con cinta adhesiva transparente. El motivo es simple, los discos HD ya valen a 66 Ptas, y los DD han desaparecido o son más caros.

Gracias Felix Alonso por tu carta "rutinasC68_txt" de mayo. Te adelanto que todo lo que pides, y mucho más, ya está presente en la nueva versión de OLIMPO de Pedro Reina para 1995. Ojalá tu interés por el C68 y el sistema OLIMPO no decaiga.

El 23 de septiembre recibi una carta de un tal Juanjo de Zaragoza de 21 años que estudia segundo de FP3 en la rama de electricidad. Por lo visto, un primo

suyo que trabaja en empresa BULL de ordenadores le ha regalado un QL con discos y un Z88, porque su mujer le ha obligado a hacer limpieza en su casa. Juanjo es también usuario de AMIGA, pero dice que ha obtenido mi dirección por mediación de gente que conocia a su primo, y que le han dicho que yo editaba un fanzine en disco de 3.5".

Cartas como la de Juanjo me hacen pensar que el futuro de QLIPER pasa por el reciclaje de los viejos QLs. En otras palabras, esos QLs olvidados y almacenados en algún lugar (espero que no hayan caido en manos del camión de la basura) deberian caer en manos de gente interesada en utilizarlos. El QL siempre ha sido una máquina que ha despertado el interés de aquel que se pone a pulsar sus teclas. Que es todo lo contrario a cuando nos ponemos delante de un PC cualquiera del montón.

La media de edad del Club QLIPER está en torno a los 45 años. Yo cuando el Club de Usuarios QL fue creado/fundado, tenia 20 años, ahora tengo 30 años. Hace 10 años todo me parecia más fácil, y tenia ilusión para hacer muchas cosas, hoy ya he perdido el entusiasmo de cuando uno es joven. Los nuevos usuarios como Juanjo (21) y Joaquin (17), tenian en aquel entonces 11 y 7 años respectivamente. Ellos tienen la juventud e ilusión, nosotros solamente la experiencia de 10 años.

Con toda la anterior historia, lo que intento comunicar es que lo que nuestro Club QLIPER necesita es usuarios con una edad de 16-21 años con muchisimas ganas de aprender y probar cosas nuevas. Sin embargo, el obtener socios que estaban aprendiendo a leer cuando el QL nació, es algo verdaderamente dificil, pero he visto milagros más dificiles en el mundo QL en el pasado.

Sobre el fin de QLIPER, he de advertir que el Club de Usuarios QL Español murió oficialmente en la primavera de 1988 cuando Serafín Olcoz dimitió, QLIPER es simplemente una metamorfosis del Grupo Local QLAVE MALAGA compuesta originalmente por socios QLAVE a nivel nacional. Mi idea en aquel entonces era continuar con la labor de QLAVE, pero solamente con un máximo de 12-20 socios/colaboradores que a su vez estarian comprometidos en recompilar colaboraciones y a copiar la revista en disco a otros usuarios que a su vez estarian comprometidos con ese socio en lo mismo (Y asi teóricamente, hasta el infinito). Tengo pruebas o indicios de que hubo resultados positivos durante los primeros años y la revista llegó a manos de más de 60 usuarios QL (60 han sido el minimo de lectores/colaboradores de nuestra revista, según nuestra antigua base de datos). Actualmente, dado el silencio de la mayoria de nuestros socios actuales y el desinterés en continuar en contacto de los antiguos socios, no sé si se continua la tradición de copiar la revista a nuestros amigos con QL.

Mi idea actual sobre QLIPER es reconvertirlo para que sea de interés para los programadores/hobbistas de hoy. Naturalmente, nunca será un Club grande, siempre será pequeño. Mi único interés es el estar en comunicación e intercambiar con gente con un interés en común.

Después de tanta historia sobre el fin de QLIPER, me temo que el único que puede dar un cerrojazo al club soy yo, pues estoy convencido de que si algún día decido abandonar, no voy a tener sucesor. Marcos Cruz era mi mejor delfín, pero ni siquiera el mejor es capaz de aguantar la soledad de un Club como QLIPER.

El pasado 17 de septiembre estuve en la nueva casa de Pedro Reina en Madrid. No sé realmente cómo conseguí llegar alli sin conocer Madrid y sin un mapa, y a las diez en punto como prometi, quizás alguien de arriba guiaba mis pasos y me daba protección, porque ni habia circulación de coches por sus calles ni gente por sus aceras, y en los establecimientos en los cuales entré para preguntar por la calle probablemente me tomaron por un policia de paisano (dicen que la gente del interior no es tan hospitalaria como en la costa). En fin, no importa todo lo anterior, pero por fin pude conocer a Pedro, su novia, su nueva casa y sus ordenadores. Alli estuve prácticamente casi toda la mañana hablando con él y viendo sus nuevos programas y actualizaciones de OLIMPO para 1995. ¡Ojalá tuviesemos más socios como él!
Pedro me pidió que os escribiese sobre mi último trabajo, el ALTAIR eFORTH. La verdad es que ha sido un trabajo muy fácil a pesar de que he tenido que estudiar assembler Intel 80c535 y el hardware del ALTAIR 535. Podeis creerlo, o no, pero escribir ALTAIR eFORTH solamente me ha costado unos 4 días más una semana depurándolo y añadiendo nuevas palabras para compatibilizarlo con otros

FORTHs. También escribi un pequeño terminal en TURBO C para el PC con la opción de enviar ficheros y salvar ficheros binarios ejecutables en solitario. Desgraciadamente, a pesar de que hay 110 pedidos esperando desde agosto, el ALTAIR eFORTH no se comercializa aún, porque el hardware del ALTAIR 535 no ha sido terminado todavia. Yo solamente tengo una placa prototipo con un interface RS-232, pero falta una pantalla LCD, un altavoz y varios interfaces/periféricos más. En los ficheros 'eforth_intro_txt' y 'eforth_glosario_txt' podréis observar los manuales iniciales de la v1.02 del ALTAIR eFORTH.

La casa Ibercomp tiene otro proyecto con nombre sin determinar (ANTARES 68008). Consiste en un microprocesador basado en el 68008 a 16 MHz, 2 puertos RS232c, 48 lineas de entrada/salida digitales, 8 entradas analógicas, 8 salidas analógicas, video compuesto, salida de audio, interface de disquettes e interface para teclado de PC. Dispondrá de 32 Kbytes de ROM, 64 Kbytes de RAM + memoria de video (texto 136x25 con 8 colores + parpadeo). Dicen que me podrian enviar un prototipo antes de finalizar el año, y que el P.V.P. será inferior a las 20.000 Ptas. Mi eFORTH 68000 escrito en el QL podria funcionar en ese prototipo en solamente un día.

Sobre la revista SINCLAIR QL WORLD se sabe muy poco, lo único que sé por mediación de la revista IQLR es que la empresa Arcwind se encuentra en proceso negociación con otras empresas, y que IQLR es una de las interesadas en adquirir los derechos de SQLW. ¡Ojalá SQLW no caiga en manos de una empresa interesada solamente en paralizar su publicación!, porque seria un duro golpe para el mundo QL el perder su mayor medio de comunicación (30.000 subscriptores).

Es de suponer que a la mayoria de nosotros, se nos hace demasiado larga la espera de nuestra revista en disco QLIPER, pero la dura realidad es que hay que reducir costes, y además, el número real de socios colaborando en la revista se ha reducido demasiado. Sin embargo, todavía somos capaces de llenar nuestro disco sin tener que tomar prestado material del mercado internacional. Aunque he de reconocer que hasta yo mismo estoy asombrado de nuestras posibilidades reales. No sé realmente qué nos espera el próximo año, pero confio en poder daros más de una sorpresa, porque se celebra nuestro décimo cumpleaños como CLUB.

¡FELIZ NAVIDAD Y PROSPERO AÑO NUEVO!

Salvador Merino, 29 de septiembre de 1994

Una última advertencia, las versiones MS-DOS de los programas de Pedro Reina están infectados por un VIRUS informático que no sabemos realmente de cual se trata, pero si vais a utilizarlos, utilizar anti-virus, o mejor aún, volver a compilar los ficheros fuente con TURBO C++. Yo os he enviado la versión original por si os interesa hacer alguna gamberrada el día de los Santos Inocentes (P.e.: Instalar un VIRUS en el ordenador del vecino).

EN ESTE DISCO
----

qdos_canales_bas	Autor: Felix Alonso	
rutinasc68_txt	"	
- Agenda	Autor: Pedro Reina	
PresentoAgenda_txt	" "	
Agenda_txt	" "	
Agenda_bas	" "	
Agenda_exe	" "	
Ejemplo_cnf	" "	
DeskJet_cnf	" "	
Agenda_c	" "	
Agenda.exe	" "	Infectado con VIRUS BARROTES
- Anarit	" "	
PresentoAnarit_txt	" "	
Anarit_txt	" "	
Anarit_bas	" "	
Anarit_exe	" "	
Anarit_dbf	" "	

```
Anarit_mak                              "      "
Anarit_h                                "      "
Anarit_c                                "      "
Problema_c                              "      "
Nucleo_c                                "      "
Anarit.exe                              "      "  Infectado con VIRUS BARROTES
**** ARTICULOS VARIOS  ***
qxl_txt                             Autor: Salvador Merino
scanner_logithec_txt                           "       "
VideoBlaster_txt                               "       "
** Foto-DBase MS-DOS SVGA **
msdos_FOTO.EXE                                 "       "
msdos_DEMO.FXI                                 "       "
msdos_DEMO.FXD                                 "       "
msdos_DEMO.DBF                                 "       "
foto_dbase_txt                                 "       "
*** ALTAIR eFORTH ***
intro_txt                                      "       "
eforth_txt                                     "       "
*** Editorial  ***
oferta_txt
inventario_txt
qliper52_scr                       Portada (lbytes flp1_qliper52_scr,1024x128)
qliper_52_txt
```

LENGUAJE C-68
=============

En principio, expresar el agradecimiento a Salvador Merino y a Pedro Reina, por sus importantes aportaciones al desarrollo de este lenguaje para nuestro QL.
A Salvador Merino, por sus programas FOTO_DBASE y MERINO_TIL.
A Pedro Reina por su programa OLIMPO, programación orientada al objeto.
Y a ambos, por sus varíados desarrollos y colaboraciones en este campo.
A mí, particularmente, me parecen también interesantes las siguiente rutinas trasladadas al C-68 por Salvador Merino, y que aparacen en en uno de los anteriores QLIPERS:

```
struct  PRESHAPE
int makeshape()
getimage()
putimage()
getpixel()
```

Asimismo me parecen interesantes las siguientes funciones que se contienen en los ficheros de encabezamiento conio.h y graphics.h, y que agradecería fueran también trasladadas al C-68, si es posible:

```
int gettext(int left,  int top,  int right,  int bottom,
void *buf)
```

Esta función copia en buf el texto que hay en el rectángulo formado por los ángulos left, top (superior izquierdo) y right, bottom (inferior derecho). Las coordenadas están en relación con la pantalla. La función devuelve 1, si no ha encontrado error y 0, en caso contrario.

```
int  movetext(int left, int top,  int right, int bottom,
int newleft, int newtop)
int puttext(int left,  int top,  int right,  int bottom,
void *buf)
```

Esta función copia un texto, previamente salvado con getttext(), desde la zona de memoria apuntada por buf, hasta la pantalla en la posición left, top (ángulo superior izquierdo) y right, bottom (ángulo inferior derecho). Las coordenadas se establecen con respecto a la pantalla.

void putpixel(int x, int y, int color)
Esta función escribe en el pixel x, y, el color especificado por el argumento color.

Félix Alonso
Burgos, Mayo 1994

PRESENTO AGENDA
================

Os traigo hoy el programa Agenda, que no es más que una adaptación de un programa que yo venía usando hace bastante tiempo.

Las agendas electrónicas y los programas de ordenador que las simulan no me gustan mucho. Prefiero apuntar en una hoja de papel lo que tengo que hacer (que es poco).

Así que preparé un programa en SuperBASIC que imprimía en mi DeskJet 500 un mes cualquiera dejando un cuadrito para cada día. Lo he venido usando mucho tiempo; lo usamos mi novia, mi hermana y yo.

Se me ocurrió hacerle algunas mejoras, para lo que decidí que lo mejor era reescribirlo usando Olimpo, ya que así iba a tardar menos y además os lo podría enseñar. Con un poco más de trabajo, he intentado que os resulte posible configurar el programa para vuestras impresoras.

Sigo investigando en la mejor manera de definir los códigos de impresora. En el programa Mondrian usé un método que consistía en escribir los códigos en decimal. Por ejemplo, para escribir la orden de expulsión de página en mi DeskJet tenía que escribir

27 38 108 48 72

esto es lo que normalmente se escribe como ESC &l0H. Como muchos de los códigos de las impresoras consisten en el ESC (código 27) seguido de varios caracteres imprimibles, pensé que el método de escribir los códigos decimales era muy poco natural y decidí cambiarlo.

Por eso en Agenda los códigos se escriben tal cual, y para poder representar fácilmente los códigos no imprimibles, uso la notación hexadecimal; así, la orden anterior la puedo escribir como

\1B &l0H

Ahora estoy más a gusto, de modo que quizá introduzca este sistema en Olimpo.

Las instrucciones del programa las tenéis en el fichero Agenda_txt; he escrito un pequeño lanzador en SuperBASIC que es Agenda_bas; el ejecutable QDOS es el fichero Agenda_exe, en formato MS-DOS es Agenda.exe; podéis ver un ejemplo de fichero de configuración en Ejemplo_cnf y he introducido el fichero de configuración que uso yo, DeskJet_cnf; por último, podéis adaptar el programa a vuestro gusto puesto que el código fuente está en Agenda_c (necesitáis Olimpo).

Pedro Reina, S.10.9.1994

Agenda
======

Versión: 2.0
Autor: Pedro Reina

Fecha: V.9.9.1994

Objetivo del programa

Agenda imprime el mes deseado en una hoja para poder usarla como agenda. Se puede imprimir una quincena por cada cara (con más espacio para cada día) o bien el mes completo en una cara.

La impresora debe disponer del juego de caracteres llamado PC-8. Éste es el juego de caracteres usado habitualmente por el PC en pantalla.

Como las impresoras suelen tener varios juegos de caracteres, el usuario de Agenda debe buscar en el manual de la impresora qué órdenes hay que enviar para seleccionar el juego PC-8. Se puede configurar Agenda para que use los tipos de letra que se deseen.

Requisitos del programa

El programa puede funcionar en un QL o en un PC, y no exige ningún requerimiento especial.

Para arrancar el programa en el QL, éste debe estar en modo de alta resolución (MODE 4); hay que teclear

EXEC_W Agenda_exe

Se puede usar cualquier orden equivalente a EXEC_W, como por ejemplo EW.

En el PC se teclea

Agenda

Opcionalmente, se puede invocar el programa escribiendo como parámetro el nombre del fichero de configuración que se desea usar. Si se deseara usar el fichero de nombre "Ejemplo", habría que teclear en el QL

EXEC_W Agenda_exe ; "Ejemplo"

(Para usar esta opción es necesario que en el QL esté instalado Toolkit II).

En el PC habría que teclear

Agenda Ejemplo

Uso del programa

El programa se maneja completamente desde un único menú.

A continuación veremos el significado de cada opción del menú:

Año. El programa toma como valor inicial el año del reloj del sistema.

Mes. El programa toma como valor inicial el mes del reloj del sistema.

Destino. Dónde se desea enviar el resultado del programa. Puede ser un puerto del ordenador donde esté instalada la impresora o un fichero.

Imprime anverso. Imprime la primera quincena del mes elegido.

Imprime reverso. Imprime la segunda quincena del mes elegido.

Imprime todo. Imprime el mes elegido completo por una sola cara.

Nota: Agenda no comprueba que el año elegido sea bisiesto, de modo que considera que todos los febreros tienen 28 días.

El fichero de configuración

Cuando se invoca Agenda se puede cargar un determinado fichero que defina algunos parámetros del programa.

Si se invoca Agenda sin especificar ninguna configuración, se busca una por defecto:

En el QL, se busca el fichero Agenda_cnf en el directorio de datos establecido por Toolkit II.

En el PC, se busca el fichero Agenda.cnf en el directorio actual.

Si no se encuentra este fichero, se utilizan los valores por defecto.

Si en el fichero de configuración que se esté leyendo se deja sin definir algún parámetro, Agenda usará para ese parámetro el valor por defecto.

Los ficheros de configuraciónn son simples ficheros de texto que se pueden crear y modificar con cualquier editor de texto. Se pueden introducir cuantos comentarios se desee. Se considera comentario toda línea que comience con el carácter '*' y toda la parte de una línea a partir de la doble barra ("//"). Las líneas en blanco se ignoran.

Para configurar un parámetro hay que introducir una línea en el fichero que tenga como primera palabra el indicador de lo que se quiere configurar y a continuación el valor que se desea dar. Se pueden introducir en cualquier orden.

Algunos parámetros permiten definir los códigos que se mandan a la impresora en determinados momentos. La manera de definir los códigos es la siguiente:

Los códigos de mandan a la impresora tal como se encuentren en el fichero, es decir, si se escribe la letra 'A', se manda la letra 'A' (que tiene el código 65). Sólo hay dos excepciones:

1. El espacio en blanco se ignora.

2. Si se escribe la barra inversa, no se envía, sino que hace que los dos siguientes caracteres se interpreten como un código hexadecimal. Por ejemplo, \1B se interpreta como el código 27.

El máximo número de códigos que se pueden almacenar es 40. Si se escriben más de 40, se ignorarán todos los excedentes. El valor por defecto es no enviar nada.

Para ayudar al usuario a crear su propio fichero de configuración se incluye uno de ejemplo.

A continuación se describe cada una de las posibles entradas:

Sonido

Escribiendo "Sí" o "No", se especifica si el programa debe comenzar con el sonido conectado o no.

Destino

Valor inicial para el parámetro "Destino" del menú principal.

Preámbulo

Las órdenes que hay que mandar a la impresora al comenzar la impresión. El usuario puede usar esta opción para seleccionar las características generales de la impresión.

PreámbuloMes

Las órdenes que hay que mandar a la impresora al comenzar la impresión del mes completo a una sola cara. Se puede usar para seleccionar el interlineado que sea más conveniente. Máximo, 40 códigos.

PreámbuloQuincena

Las órdenes que hay que mandar a la impresora al comenzar la impresión de cada quincena. Se puede usar para seleccionar el interlineado que sea más conveniente. Máximo, 40 códigos.

Postámbulo

Las órdenes que hay que mandar a la impresora para terminar la impresión. Máximo, 40 códigos. El uso habitual de esta opción es volver a seleccionar en la impresora los parámetros que se utilicen habitualmente; para ello, lo normal es enviar una orden de "reset".

Cuadro

Las órdenes que hay que mandar a la impresora al comenzar la impresión de cada línea de las que componen el cuadro. Máximo, 40 códigos. Ésta es la opción que se utiliza para seleccionar en la impresora el juego de caracteres PC-8 y el tipo de letra base para el cuadro. Se utiliza también para colocar la impresora en la columna exacta para escribir los nombres de los días y del mes.

Texto

Las órdenes que hay que mandar a la impresora al comenzar la impresión del nombre de cada día. Máximo, 40 códigos. Se puede usar para seleccionar el tipo de letra.

Título

Las órdenes que hay que mandar a la impresora al comenzar la impresión del nombre del mes. Máximo, 40 códigos. Se puede usar para seleccionar el tipo de letra.

AnchoMes

El número de líneas que se asignarán a cada día cuando se imprima el mes completo.

AnchoQuincena

El número de líneas que se asignarán a cada día cuando se imprima cada quincena del mes.

FilaMes

La fila en que se imprimirá el nombre del mes cuando se imprima el mes completo.

FilaQuincena

La fila en que se imprimirá el nombre del mes cuando se imprima cada quincena del mes.

ColumnaMes

La columna en que se imprimirá el nombre del mes cuando se imprima el mes completo.

ColumnaQuincena

La columna en que se imprimirá el nombre del mes cuando se imprima cada quincena del mes.

PRESENTO ANARIT
===============

En un número anterior de QLíper os enseñé un programa llamado Cifras, del que os dije que era el que más me gustaba de todos los que había hecho. Pues bien, al programa le he lavado la cara, le he preparado un interfaz y creo que por tanto ahora se le puede sacar por ahí más fácilmente (es menos cutre de presentación).

Estos son los ficheros que podéis encontrar:

Anarit_txt: las instrucciones.
Anarit_bas: un lanzador en SuperBASIC
Anarit_exe: el ejecutable QDOS
Anarit.exe: el ejecutable MS-DOS
Anarit_dbf: la base de datos
Anarit_mak, Anarit_h, Anarit_c, Problema_c, Nucleo_c: el código fuente

Pedro Reina, S.10.9.1994

Analizador aritmético
=====================

Versión: 2.0
Autor: Pedro Reina
Fecha: S.10.9.1994

Objetivo del programa

Este analizador aritmético resuelve un problema real que se presenta en el concurso televisivo "Cifras y letras" utilizando técnicas de inteligencia artificial de búsquedas en espacios de estados. El enunciado del problema es:

A partir de 6 números obtenidos del conjunto {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 25, 50, 75, 100} (se pueden repetir) hay que obtener otro número natural perteneciente al intervalo [100,999]. Se pueden utilizar las cuatro operaciones elementales entre los números y cada número sólo puede ser utilizado como máximo una vez. El ganador es el concursante que más se aproxima al número (por defecto o por exceso).

El programa implementa una búsqueda por un árbol de estados que tiene un número máximo de aproximadamente 3 millones y medio de nodos.

Derechos

Se puede obtener el código fuente, así como conectar con el autor:

Pedro Reina
c/ Marquesa de Argüeso 4
28019 Madrid
España

Teléfono: 565 17 59

Uso del programa

El programa puede funcionar en un QL o en un PC, y no exige ningún

requerimiento especial.

Para arrancar el programa en el QL, éste debe estar en modo de alta resolución (MODE 4); hay que teclear

EXEC_W Anarit_exe

Se puede usar cualquier orden equivalente a EXEC_W, como por ejemplo EW.

En el PC se teclea

Anarit

El programa se dirige desde un menú horizontal que se encuentra permanentemente en la parte superior de la pantalla. Veremos el significado de cada opción del menú, examinando también los submenús que van apareciendo:

Salida. Permite concluir la ejecución del programa.

Explica. Para recibir distintas explicaciones acerca del programa.

Analiza. La opción principal, que permite analizar un determinado problema. Tiene cinco opciones que permiten definir un problema y todas ellas desembocan en el análisis del problema y posteriormente, en su grabación (optativa). Las opciones son:
Introducción de problema: permite definir un problema cualquiera como los que aparecen en el concurso televisivo.
Introducción de problema extendido: permite definir un problema con datos mucho más complicados que los que se admiten en el concurso. La lógica interna del programa no hace ninguna diferencia con los dos tipos de problema.
Generación de problema: el programa genera aleatoriamente un problema.
Generación de problema: el programa genera aleatoriamente un problema con datos complicados.
Lectura de problema: se puede leer un problema de una base de datos. El nombre de la base de datos se puede especificar en la opción Fichero del menú Opciones.

Opciones. Para definir el comportamiento del programa en varios aspectos. Presenta estas posibilidades:
Sonido: activa o inactiva la emisión de pequeñas señales acústicas.
Grabación: especifica cuándo se grabarán los problemas una vez están analizados; hay tres posibilidades: automática (los problemas siempre se graban), manual (en cada problema se pregunta si hay que grabarlo) y nunca (no se graban).
Fichero: señala el nombre de la base de datos que se utiliza para leer y grabar problemas.

Base de datos

El programa se entrega con una base de datos (en formato estándar xBase) que contiene varios casos reales tomados del concurso de Televisión Española. Todos ellos tienen algo en común: en el concurso no se llegó a encontrar la solución exacta.

La recopilación de los datos se debe a la paciencia de Natalia Ortega Sáez.

Génesis del programa

Este programa es fruto de una pequeña casualidad. Por un lado, me encontraba estudiando la asignatura "Conceptos Fundamentales II", impartida por Martín Molina (de "EQ sistemas inteligentes") en el curso "Sistemas expertos en la empresa" impartido en el CETICAM durante el año académico 1991-92 y por otro lado a la hora de comer me gustaba pasar un rato con el programa "Cifras y letras", de Televisión Española (por La 2).

Pronto me di cuenta de que el problema propuesto en el programa se podía resolver usando las técnicas de inteligencia artificial que explicaba Martín Molina. Me puse manos a la obra en cuanto mi manejo del C fue lo suficientemente

fiable. Por cierto, aprendí mucho C mientras escribía el programa.

Fui escribiendo el código en ratos libres, algunas veces mientras mis alumnas de Matemáticas de C.O.U. se estaban examinando.

La idea principal para implementar el programa me la ofreció el propio Martín Molina. Luego tuve que optimizar el algoritmo, lo que constituyó un magnífico ejercicio de programación real.

Lo escribí en un PC portátil (un 386sx a 20 MHz) con la intención de que en esa máquina llegara a resolver el problema en tiempo real, es decir: en los 45 segundos que se ofrecen en TV. Una vez conseguido eso, dejé de preocuparme por la velocidad de ejecución. Por lo tanto, el rendimiento puede ser muy malo en una máquina inferior (un 8088, por ejemplo) o magnífico en una máquina rápida. Este es un programa que mejora con el procesador y con su velocidad.

Durante el desarrollo del programa tuve muchos ánimos de mis profesores en el curso de Sistemas Expertos (señalo especialmente a Antonio Salmerón), de mis compañeros en el curso, de mis alumnos de educación secundaria (especialmente Natalia Ortega Sáez) y de mi novia Julia sin cuya compresión y ayuda no hubiera podido terminar nunca el programa.

MI PRIMER VIRUS EN MI PC

A pesar de que mi PC 486 utiliza preservativo (el anti-virus de Microsoft, el cual no parece ser muy bueno), y prácticamente ha tenido poco contacto con copias piratas de software, he de reconocer que me han colado un VIRUS del tipo DESCONOCIDO. Sus caracteristicas son:

- Engorda ficheros .EXE y .COM en 1310 bytes.
- Altera algo en ficheros .SYS.
- FORMAT formatea discos HD con muchos errores, pero formatea HD a DD sin errores.
- DISKCOPY se niega a duplicar un disco en la unidad A:, porque dice que es una unidad de disco no extraible.
- El fichero COMMAND.COM era imposible de borrar desde el MS-DOS, pero consegui borrarlo desde el anti-virus del WINDOWS.

He conseguido eliminar el VIRUS de mi PC borrando todos los ficheros infectados, y volviendo a configurar el MS-DOS y WINDOWS. Ahora solamente me quedan varias dudas:

- ¿COMO SE INTRODUJO EL VIRUS EN MI PC?
- ¿ESTA EL VIRUS ESCONDIDO EN ALGUN LUGAR DEL DISCO DURO ESPERANDO PARA VOLVER A REPRODUCIRSE?
- ¿ME VENDIERON EL PC CON EL VIRUS INSTALADO?

Lo único que sé, es que por la fecha en que el Virus ha empezado a reproducirse, y por los ficheros infectados, todo hace pensar que el Virus me lo ha colocado la firma IBERCOMP en el software para programar su Automata SIEMENS 80c535, pero el anti-virus de Microsoft no lo detecta.

Sobre VIRUS se ha escrito rios de tinta en el mundo PC. Es una alegria saber que el SINCLAIR QL es un ordenador inmune a virus, pero no sé si la QXL podria estar a salvo en el futuro de virus especiales para ella, ya que suministrar el software en disco para correr en RAM convierte al sistema SMSQ en vulnerable.

Salvador Merino, 3 de agosto de 1994

Por fin he conseguido aislar el VIRUS definitivamente de mi PC encontrando su raiz. El VIRUS estaba escondido en un disquette 5.25" 100% original y comercial que contenia mi eFORTH para PC.

Como veis, en el mundo PC los VIRUS más peligrosos son suministrados directamente

por las casas comerciales en sus disquettes comerciales. También hay que tener en cuenta que mi programa eFORTH data de 1990, que es la época dorada de los VIRUS.

¿Qué cómo me di cuenta?

Tenia todo el arsenal ANTI-VIRUS en alerta y funcionando a tope cuando decidi probar por última vez (antes de borrarlo) el programa eFORTH para PC instalado en el disco duro. Solamente hice ejecutarlo y el VSAFE da un pitido que me advierte de que hay un intento de modificar el COMMAND.COM. Casi al borde del infarto por la impresión, selecciono la opción detener. Luego me advierte de un intento de modificación de la memoria, sigo seleccionando la opción detener. Salgo del eFORTH con un BYE, y hago un DIR para ver si el COMMAND.COM ha sido modificado. Como veo que no, lo siguiente que hago es borrar el eFORTH.COM del disco duro y BYE-BYE al indeseable VIRUS.

Salvador Merino, 9 de agosto de 1994

QXL, POR FIN TERMINADA AL 100%

La semana pasada recibi, con grata sorpresa, la última actualización del software de la QXL. La novedad era la incorporación del nuevo SBASIC.

SBASIC tiene varias diferencias frente al veterano SuperBASIC, las cuales vienen explicadas en el pequeño manual de 6 páginas. La mayor ventaja del SBASIC frente al SuperBASIC STANDARD (no MINERVA) es la posibilidad de tener tantos SBASIC ejecutándose como deseemos o la memoria nos permita.

Con esta entrega fechada 6 de junio de 1994, podemos afirmar que el software y hardware QXL se encuentran en un nivel 99% operativo. Ya lo único que queda es reescribir algunas rutinas lentas por otras más rápidas e ir depurando posibles errores a medida que se vayan descubriendo.

Algo que me gustaria obtener, es un manual de desarrollo de aplicaciones para la QXL en el que viniese explicado cómo acceder al hardware de un PC desde la QXL (especialmente tarjetas como Video Blaster, SOUNDBLASTER, SCANNERS,...), pues me permitiria, entre muchas cosas, utilizar modos gráficos de muchos colores no soportados por el sistema operativo SMSQ.

El manual de la QXL aún no ha sido suministrado. Todo hace entender que las tarjetas QXL han sido vendidas a solamente usuarios de QL, pues con el contenido de los ficheros de texto que se suministran en el disco, poquita cosa se puede hacer. Sobre el sistema operativo SMSQ, excepto su compatibilidad con el QDOS, se sabe muy poco. Para escribir software que utilice las ventajas que ofrece el nuevo hardware, HACE FALTA DOCUMENTACION.

Actualmente la única alternativa capaz de ampliar el parque de QLs, es la QXL. Sin embargo, seria necesario una campaña de publicidad, que pudiese convencer al usuario final, unida a una colección de nuevos programas que hiciesen el mejor uso posible del sistema operativo SMSQ. Si he de ser sincero, he de advertir que es muy raro que un usuario de PC se decida por comprar una QXL, pues con los programas MS-DOS y WINDOWS que piratea, tiene suficiente para lo que normalmente usa el ordenador, pero de lo que si estoy seguro es que los usuarios de QL incondicionales, tarde o temprano, van a comprar un PC y una QXL.

Salvador Merino, 2 de Agosto de 1994

QXL, UN AÑO DESPUES

Aunque el hardware de la tarjeta QXL está terminado, y es bastante fiable, el software QXL no está terminado aún en un estado minimo de emulación de un SINCLAIR QL, pues falta el nuevo BASIC compatible SuperBASIC.

Yo pedi mi QXL 2M justo cuando vi el primer anuncio en la revista SQLW, pero no la recibi hasta finales de agosto. Luego hubo que comprar un PC 486 que no he

podido disfrutar hasta hace un par de meses, pues como ya os he contado, me comprometí con un pedazo de inútil, o sinvergüenza total, que con tanto mañana - mañana y en 24 horas todo solucionado, me ha tenido liado y esperando durante 8 meses (y más tiempo podria haber sido si no hubiese decidido por mi cuenta reparar y sustituir las piezas defectuosas del PC a cuenta de mi sufrido bolsillo).

MI MEJOR CONSEJO ES QUE DEBEIS EVITAR EN ESPAÑA PAGAR ALGO POR ADELANTADO SIN VERLO Y PROBARLO ANTES, porque hasta una garantia de 2 años podria tratarse de papel mojado. Por lo pronto, la casa STRONGER es altamente NO recomendable.

¿Qué tiene el SMSQ v2.10?

Drivers: pantalla (QL, EGA, VGA y SVGA), teclado (Inglés, Francés y Alemán), SER, PAR, RAM, FLP, WIN y QL-NET.

Como habreis observado, no hay driver de teclado Español, este texto lo estoy escribiendo con mi QXL 5M y la ayuda de una chuleta donde he escrito todas las combinaciones de teclas para los caracteres Españoles y otros que no coinciden con el teclado Español. En otras palabras, todo un engorro e incomodidad, pero superable. Sin embargo, el PC trabajando con teclado Español bajo MS-DOS o WINDOWS no está exento de problemas, porque los caracteres Ingleses que han sido sustituidos por los Españoles, no hay manera de obtenerlos con la mayoria de los editores de texto (p.e.: Para poder obtener el carácter '~' que sirve para negar en el lenguaje C, me he visto obligado por sencillez a utilizar la QXL, y ahí no queda todo, cualquiera de los editores de texto del QL es mejor que todos los editores de texto que poseo para PC).

[El problema de utilizar caracteres que no vienen en el teclado se soluciona en MS-DOS pulsando ALT y el código ASCII. En modo QXL se soluciona pulsando la combinación correcta de teclas]

En mi configuración, tengo un disco WIN1_ de 10 Mbytes con un BOOT que carga y ejecuta el QPAC I y II desde disco duro.

Dos programas que siempre se han utilizado para comprobar la compatibilidad QL han sido siempre el paquete de gestión de PSION y el CHESS de PSION. Yo uso con mi QXL 5M las versiones para el THOR si ninguna dificultad.

Otro programa que tengo en disco duro es el compilador C68 v.4.01, el cual compila muy rápido desde disco duro (¡Ojo! Desde FLP es una tortuga). Y lo bueno que tiene es que los programas C68 son 100% compatibles con la QXL, y no solamente eso, escribir programas para el entorno QPTR es bastante más fácil en C68 que en otros lenguajes QL (SuperBASIC o Ensamblador).

En esta versión del sistema operativo SMSQ es posible compilar programas SuperBASIC con el compilador QLiberator última versión. También he comprobado que programas SuperBASIC compilados con versiones antiguas del QLiberator corren bien si utilizamos el runtime actual y programa modificador suministrado con la ROM MINERVA.

También he probado algunos programas SuperBASIC compilados con TURBO y corren bien.

También he comprobado que versiones antiguas de programas comerciales bajo el entorno QPTR de Tony Tebby no corren bien, pues han sido escritos utilizando versiones del QPTR obsoletas, pero sé que sus versiones actuales corren estupendamente.

Naturalmente, el viejo SuperFORTH v2.0 de D.P., también corre estupendamente. Realmente hay muy pocos programas EXECutables que no corran en la QXL, y esos pocos deben tratarse de programas que acceden directamente al hardware del QL sin pasar por las llamadas al sistema operativo, o simplemente contienen errores que pasan inadvertidos en un SINCLAIR QL dependiendo de la versión de su ROM.

Actualmente la tarjeta QXL puede instalarse en ranuras ISA de 8 y 16 bits (preferible de 16 bits), y el problema del formateo de discos ha sido solucionado. El único inconveniente es que el acceso y velocidad de transferencia del FLP es demasiado lento (esas rutinas necesitan ser revisadas y actualizadas).

Hace un par de meses MIRACLE SYSTEMS ha lanzado al mercado la SUPER GOLD CARD, y ha eliminado la publicidad de la tarjeta QXL temporalmente de la revista SQLW, pero no de la IQLR. Yo creo que como es una casa seria, ha decidido no continuar aumentando el parque de QXLs hasta que el software esté totalmente terminado. Las unidades vendidas sirven para comprobar la compatibilidad con el software actual, y para que los programadores escriban su nuevo software o actualizaciones de viejos titulos guardando compatibilidad con la QXL y las otras alternativas QL-compatibles (AMIGA-QDOS, ATARI/QL, MINERVA, SMS-2,..).

La SUPER GOLD CARD es un producto apetitoso para continuar con el viejo SINCLAIR QL, pero dado el mercado actual dominado por los PCs y que el parque de QLs es de serie limitada (solamente para coleccionistas y usuarios de ordenadores raros), la QXL es la verdadera alternativa de futuro. Pero debo advertir que a los usuarios de PC le importa la multitarea y la programación un pito, y ahora lo que mola es correr programas WINDOWS, pues menear el ratón en un entorno gráfico con muchos colores es lo que está de moda. Sin embargo, nosotros tenemos nuestro equivalente a WINDOWS, el QPAC, que incluso ya existia en nuestros QLs antes que el WINDOWS fuese un proyecto, pero tiene la pega de que tiene actualmente todavía la limitación de 4 colores (2 colores en el SMS-2 del ATARI).

La QXL es una buena alternativa para competir con sistemas operativos multitarea como el OS/2, UNIX o el misterioso WINDOWS NT. En un principio la QXL podria tener un precio caro comparado con los nuevos precios de un OS/2 o un UNIX pequeño, pero necesita muchisimo menos hardware para funcionar que su equivalente en el mundo Intel 80x86, y hay que tener en cuenta que lleva el hardware de red local. Si se juegan bien las cartas, Miracle Systems podria encontrar el pequeño mercado que necesita para sobrevivir. Sin embargo, con la nueva guerra comercial que se avecina, que promete ser muy encarnizada, con la entrada en acción de los PowerPC de IBM y APPLE, todo hace indicar vamos a tener máquinas RISC de 64 bits en nuestros escritorios antes de fin de siglo XX, pero se han vendido tantos PCs que veo muy dificil su desaparición del mercado a largo plazo (El QL ha resistido en peores condiciones durante 10 años).

Salvador Merino, 19 de mayo de 1994.

LOGITHEC SCANMAN 32 GRISES CON OCR Y SOFTWARE WINDOWS

Por solamente 19.995 ptas me he comprado en una tienda de informática, que hay al lado de mi casa, un Scanner de mano Logithec. Sin ir más lejos, el más bajo de la gama Logithec, pero mi pregunta es: ¿Necesita el usuario de casa, o incluso oficina, un Scanner de mayores prestaciones?. Mi respuesta es no, pues este Scanner de 400 dpi de resolución ofrece unas prestaciones 4 veces superior a mi impresora EPSON de 24 agujas (360x180 dpi). En otras palabras, digitalizas en 32 grises una fotografia, y luego necesitas una impresora láser (o chorro tinta de alta resolución) como minimo para que puedas ver lo mismo que has digitalizado a 100 dpi impreso por la impresora en un papel a 300 dpi.

Con el Scanner se suministra el software SCANMAN, FotoTouch Color y Omnipage Direct. Todos cumplen la norma TWAIN. Se suministran con los manuales en Castellano haciendo un total de unas 425 páginas.

SCANMAN es el software que controla el Scanner. Es posible digitalizar en vertical (de arriba a abajo) y Horizontal (de derecha a izquierda y izquierda a derecha). El ancho máximo para escanear es de 13 cm y regulable, pero es posible escanear una imagen por trozos y el software se encarga de unirlos automáticamente. Se puede digitalizar en grises en resoluciones de 100 dpi a 400 dpi, y también en lineas (blanco y negro). Los archivos se guardan en formato TIFF, PCX, EPS y BMP.

FotoTouch Color es un programa de tratamiento de imágenes. Con este programa se puede retocar la imagen e imprimirla en papel.

OmniPage Direct es el programa reconector óptico de caracteres. Necesita un

editor de texto (p.e.: el WRITER del WINDOWS) o una hoja de cálculo. El reconocimento se hace en ficheros digitalizados en blanco y negro a 300 dpi, y puede hacerse por columnas sencillas de texto, páginas completas, y columnas y tablas.

Con el OCR he hecho muchas pruebas y he obtenido resultados muy buenos y otros desastrozos. La mejor manera de obtener buenos resultados es practicar. De donde se obtienen muy buenos resultados es en periódicos, revistas y libros que posean letra de buena calidad. La calidad LQ de mi impresora EPSON es la calidad minima que acepta. He de advertir que incluso mi mejor prueba no ha estado exenta de un sólo error (un carácter no reconocido).

En resumen, si el Bejamín es una maravilla, ¿cómo serán los Scanners de gama alta?

Salvador Merino, 20 de mayo de 1994

VideoBlaster (Una capturadora de video para PC)

A pesar de que estoy contento con las prestaciones de mi tarjeta capturadora de video Video Blaster, he de reconocer que sigo estando enfadado, porque no pude disfrutar de ella hasta el octavo mes desde la fecha de compra, y gracias a que decidi enviarla a reparar al fabricante CREATIVE en Singaphur, pues aqui en España me dejó tirado la tienda y el almacenista-distribuidor, ya que no solamente no fueron capaces de repararla o sustituirla por una nueva, sino que encima me quemaron los manuales con cigarrillos y me robaron el disco demo CD-ROM del programa Video For Window.

Los manuales de la Video Blaster vienen en Inglés. La mayor parte del software viene para WINDOWS, excepto unos pequeños programas MS-DOS que permiten configurar el sistema y capturar imagenes.

Instalar la Video Blaster es muy fácil, pero mucho OJO, la tarjeta de VIDEO SVGA tiene que tener un "FEATURE CONNECTOR" para conectar interiormente ambas tarjetas mediante un cable plano. Por lo que sé, el 90% de los PCs vendidos en España tienen instaladas tarjetas de Video SVGA que no tienen "Feature Connector".

Las prestaciones de mi Video Blaster son buenas, pero limitadas a 640x480 pixels. Las versiones actuales de la Video Blaster están limitadas a 800x600 pixels de resolución máxima, y el software de regalo es muchisimo mejor en calidad y prestaciones en comparación con el suministrado en mi vieja versión de mi Video Blaster.

Las capturadoras de Video en comparación con los Scanners tienen menor resolución por punto de pulgada. Aunque también es cierto que los video domésticos no poseen de mucha resolución que digamos (la resolución de una fotografia sigue siendo superior a un Video doméstico). Sin embargo, las prestaciones de una capturadora de video actual son excelentes con sus 16 bits de colores de una paleta de un par de millones.

Cuando tenia la mitad de mi disco duro vacio, pude entretenerme con Video For Windows en probar sus posibilidades para crear mini/peliculas. Ahora que mi disco duro se ha quedado reducido a 33 miserables Megas libres, solamente puedo capturar fotogramas sueltos. Para utilizar Video For Windows es necesario tener un disco duro de 300 Megas como minimo con 100 Megas libres. Y os advierto que se os quedará pequeño en seguida, porque cualquier programa Windows se chupa de 10 a 20 Megas del disco duro solamente en su instalación.

El hardware de la Video Blaster está muy lejos de un QL Real Time Digitalizer o un digitalizador de video de Spem, pero lo que se obtiene en un QL se puede imprimir sin problemas en una impresora de 8 o 24 agujas, y en muchas de color, y en cambio, lo que se obtiene en una Video Blaster se necesita una impresora capaz de imprimir millones de colores por pixels.

Si estais interesados en una capturadora de Video, y vuestro PC tiene en su tarjeta de Video el famoso "Feature Conector", os recomiendo que busqueis y rebusqueis en el mercado, porque hay muchisimas mejores opciones que la Video Blaster y a mejor o igual precio.

Salvador Merino, 23 de septiembre de 1994

FORTH

AUTOR

Mi nombre es Salvador Merino. Soy socio de Forth Interest Group de California desde 1986 (Organización a la cual debo todos mis conocimientos sobre el FORTH).

El FORTH ha sido siempre un lenguaje que me ha llamado mucho la atención por su sencillez. He escrito versiones en Assembler para el Cambridge Z88 y el Sinclair QL, e incluso una versión en ANSI C STANDARD que corre en ordenadores QL y PC.

Como todos los lenguajes, ALTAIR eFORTH está sujeto a cambios y mejoras, pues se encuentra en continuo desarrollo. Cualquier usuario que pueda demostrar que es usuario registrado (que ha comprado el lenguaje legalmente), puede ponerse en contacto conmigo por carta a la siguiente dirección:

Salvador Merino
Ctra Cádiz, Cerámicas Mary
29640 Torreblanca del Sol
Fuengirola (Málaga)

PREFACIO

FORTH es un lenguaje que fue muy popular entre los microordenadores de finales de los 70 y principios de los 80. La razón era sencilla: es un lenguaje fácil de utilizar y corre más rápido que el BASIC interpretado.

FORTH constituye un puente entre quienes prefieren los lenguajes de "alto nivel" (como BASIC, PASCAL, C,..) y quienes necesitan una velocidad de ejecución como la que proporciona el Assembler. De hecho, es un lenguaje "ensamblador de alto nivel", con todas las ventajas de un ensamblador y ninguno de sus dolores de cabeza, siendo mucho más fácil de aprender y utilizar.

FORTH es muy diferente a la mayoría de los lenguajes, por su forma de escribirlo y su modo de desarrollarlo. En vez de escribir instrucciones en la forma acostumbrada, conocida como "Notación Infija" (por ejemplo, 5 + 6 + 7) utiliza la "Notación Polaca Invertida" (por ejemplo, 5 6 + 7 +).

El FORTH no sólo utiliza una notación peculiar, también es un lenguaje basado en el uso de la pila. La mayoria de los lenguajes utilizan variables para albergar los valores en curso, y "parámetros ficticios" para representar argumentos para subrutinas; el FORTH utiliza la pila para esas tareas.

El FORTH tiene variables, pero se utilizan poco en comparación con la utilización en otros lenguajes. El utilizar una pila en lugar de variables no causa grandes problemas. Un compilador (el PASCAL) normalmente tiene una pila que usted no ve y que trabaja por usted; pero el FORTH le deja a usted ese trabajo.

A pesar de que el FORTH compila sus programas, el desarrollo es tan simple como en el BASIC. La edición, compilación y ejecución de programas están integrados en una sola fase.

El FORTH implementado en el ALTAIR 535 sigue el eFORTH STANDARD, el cual es un FORTH diseñado especialmente para microcontroladores.

eFORTH Y figFORTH

eFORTH es el modelo de un FORTH diseñado para ser portable a un gran número de nuevos procesadores más potentes, los cuales están disponibles actualmente, o lo estarán en el futuro.

Forth Interest Group creo el modelo figFORTH en los años setenta para 6 microprocesadores diferentes. El problema más serio que posee el modelo figFORTH es que asume que el microprocesador anfitrión (P.e.: el PC, ATARI ST, QL, AMIGA) tiene que ser de 8-bit, y la pila de 16 bits.

En los años setenta, un ordenador personal tenia que ser autosuficiente. En otras palabras, tenia o necesitaba una CPU con memoria, teclado, monitor, unidad de disco, y un sistema operativo incluyendo lenguajes y utilidades.

figFORTH, como otros sistemas FORTH de su época, tenia que ser autosuficiente también. Tenia que soportar entrada de teclado, salida a monitor e impresora, y unidad de disco como sistema de almacenamiento. Un editor era siempre la primera utilidad FORTH que un usuario necesitaba.

F83 fue un paso de gigante. Mejor editor, más herramientas de depuración, y un interface más amigable con el sistema operativo. Sin embargo, F83 sigue siendo de un sólo usuario utilizando una única máquina.

El cambio más grande en los años ochenta fue la separación del procesador del sistema de desarrollo de aplicaciones, con la proliferación de los PCs compatibles y herramientas software para programar otros procesadores (existen herramientas de desarrollo para programar otros procesadores ya sea en PC como en ATARI ST, AMIGA, MAC, QL,...). La mayor parte de las actividades de programación de nuevos y más avanzados procesadores está siendo realizada hoy en ordenadores anfitriones separados de los ordenadores destino. Por ejemplo, el ALTAIR eFORTH ha sido desarrollado en un PC, utilizando un ensamblador 8051 y un debugger, y probándolo en un prototipo del ALTAIR 535 via RS-232.

La separación del sistema de desarrollo de aplicaciones del ordenador destino tiene un gran impacto en el desarrolo del eFORTH, el cual presume de ir dentro del procesador destino, y no en el procesador anfitrión. Es también un gran avance en el diseño de eFORTH, porque eFORTH no tiene que incluir cosas soportadas por el ordenador anfritión, y los requerimientos de eFORTH pueden ser drásticamente simplificados. En un sentido, eFORTH está direccionando las necesidades de los procesadores EMBEDDED, es el último grito en el mundo del microprocesador.

El microordenador para el cual figFORTH fue diseñado contenia una CPU, alguna memoria, un teclado, un monitor y una unidad de disco. En cambio, el microordenador eFORTH es mucho más simple. Contiene una CPU con alguna memoria, y probablemente algunos periféricos sin especificar. Está conectado al mundo exterior a través de un cable RS-232. Esta linea RS-232 probablemente no será utilizada en aplicaciones reales, pero es necesaria para conectar al ordenador anfitrión para programar, comprobar y depurar.

INTRODUCCION

Una vez encendido y conectado el ALTAIR 535 con una EPROM conteniendo eFORTH, a un PC con el programa terminal ejecutándose. Lo primero que vemos es el mensaje de Copyright:

ALTAIR eFORTH v1.02
Salvador Merino, 1994

Si deseas ver todo el vocabulario del eFORTH, teclea:

WORDS <ENTER>

Los nombres de 200 palabras/comandos van a aparecer en la pantalla.

La mayoria de las palabras eFORTH están en mayúsculas. He de advertir que es carácter sensitivo.

Escribe la siguiente secuencia de números:

1 2 3 4 5 6 5 <ENTER>

eFORTH devolvera un 'ok' al final de la linea. Estos números están almacenados en la pila. Para ver el contenido de la pila, teclea:

.S <ENTER>

Y verás en la siguiente linea el contenido de la pila:

1 2 3 4 5 6 7 <sp ok

Con estos números en la pila, teclea algunos comandos aritméticos como:

+ * .S <ENTER>

Y verás:

1 2 3 4 65 ok

Números y palabras deben estar separados por espacios. Si intentas ejecutar dos palabras juntas sin espacios, como:

123+456 <ENTER>

eFORTH devuelve:

123+456 ?

Eso significa que no entiende 123+456, porque no es ni un número ni un comando.

DUMP es útil para examinar una porción de la memoria. Teclea lo siguiente:

HEX 8000 DECIMAL 100 DUMP

DUMP muestra el contenido de los 100 bytes siguientes a la dirección HEX 8000.

SEE permite decompilar una palabra compilada en el diccionario, teclea:

SEE DUMP <ENTER>

Y verás los tokens que componen DUMP mostrados en la pantalla. Pulsa ENTER dos veces para parar, porque no para al final de la lista de DUMP.

Para definir un comando:

: PRUEBA FOR R@ . NEXT ; <ENTER>

Escribe WORDS, y verás que PRUEBA ha sido anexionado al diccionario. Para probar PRUEBA, teclea:

10 PRUEBA <ENTER>

y verás:

10 9 8 7 6 5 4 3 2 1 0 ok

Otro ejemplo de bucle:

```
: HOLA  BEGIN ." Hola mundo" CR ?RX UNTIL DROP ;
```

La palabra HOLA muestra en Hola mundo linea tras linea hasta que pulsamos una tecla cualquiera. Es un ejemplo de bucle BEGIN......UNTIL

La palabra CONSTANT no ha sido definida en el vocabulario eFORTH STANDARD. No es necesaria, porque es más rápido en ejecución definir una constante como una palabra normal:

```
: DOS 2 ;
```

Sin embargo, si desean utilizar CONSTANT, la definición de CONSTANT es:

```
: CONSTANT CREATE , DOES> @ ;

2 CONSTANT DOS
```

EL TERMINAL PARA PC

Se trata de un terminal muy simple. Lo único que hace la mayor parte del tiempo es enviar al ALTAIR todo lo que se pulsa en el teclado del PC, e imprimir en pantalla todo lo que se recibe del ALTAIR. También posee 3 funciones u opciones extra:

- CONTROL-S .- Sale del terminal al MS-DOS.

- CONTROL-E .- Abre un fichero texto conteniendo un programa FORTH y lo envia al ALTAIR como si se estuviera escribiendo en ese momento en el teclado del PC.

- Una opción de recibir una fichero binario desde el ALTAIR. Se activa automáticamente si la palabra ALONE se ha ejecutado correctamente.

PALABRAS NO INCLUIDAS EN EL eFORTH STANDARD, PERO DEFINIDAS POR COMPATIBILIDAD

FORGET   (----).- Sirve para borrar palabras del diccionario. Por ejemplo FORGET HOLA borra HOLA, pero también todas las palabras que se han definido después de HOLA.

DOES>   (-----).- Muy interesante para la creación de estructuras de datos complejas. Se utiliza con CREATE.

```
: nombre_de_tipo
         CREATE
               acción en tiempo de compilación
               DOES>
                     acción en tiempo de ejecución
                                                 ;
```

DO .... LOOP (limite indice ---).- Es un bucle que repite su interior hasta que consume su limite e indice suministrado.
LEAVE sirve para abortar el bucle.
I deja el valor de indice actual.
J deja el valor de indice actual del bucle DO ... LOOP exterior.

DO ..... +LOOP .- Es idéntico al anterior, pero con la diferencia de que el indice se incrementa con un valor que va antes de +LOOP. Por ejemplo:

```
:  ....  10 0 DO ...... 2 +LOOP ;
```

PALABRAS ESPECIFICAS ALTAIR 535

ESCRIBIR_PHANTOM_WATCH
(año mes día día-semana hora minutos segundos centésimas-segundo ---)
Ajusta el reloj/calendario del sistema. Los datos deben estar en base HEX, pero se escriben como si estuvieran en base decimal.

LEER_PHANTOM_WATCH
(--- cetésimas-segundo segundos minutos hora día-semana día mes año)
Lee el reloj/calendario del sistema. La base debe estar en HEX.

I! (byte dirección_interna ---) .- Guarda un byte en una dirección interna.

I@ (dirección_interna --- byte).- Deja el byte almacenado en una dirección interna.

CREANDO UNA APLICACION SOLITARIA

ALTAIR eFORTH ha sido diseñado con la característica de poder realizar aplicaciones finales. En otras palabras, aplicaciones finales grabadas en EPROM que se ejecutan cuando el ALTAIR 535 es encendido.

Un ejemplo podria ser el siguiente:

```
: HOLA !IO BEGIN ." HOLA MUNDO" CR AGAIN ;   <ENTER>

ALONE HOLA <ENTER>
```

La palabra ALONE lo que hace es enviar un fichero binario al PC de la aplicación que ejecuta HOLA. Una vez recibido el fichero en el PC, este pregunta por el nombre de un fichero para guardarlo en disco. Ese fichero binario hay que almacenarlo en una EPROM a partir de la dirección HEX 100 sin olvidar de grabar el sistema operativo del ALTAIR.

Nuestro ejemplo HOLA, lo único que hace es enviar por el interface serie el mensaje "HOLA MUNDO" y retorno de carro/nueva linea indefinidamente.

Advertencia: !IO lo que hace es inicializar el dispositivo serie.
Las aplicaciones solitarias en EPROM necesitan 32 Kbytes de memoria RAM para volcar el contenido de la EPROM $8000 posiciones más arriba, ya que el sistema FORTH trabaja con direcciones absolutas y las variables del sistema, variables del usuario y las dos pilas ocupan muchisima más memoria de la que ofrece la memoria interna de un Intel 80c535.
Los 2 últimos Kbytes de la memoria RAM pueden ser utilizados por el sistema para almacenar datos/variables, pero tenga cuidado con los dos últimos bytes si usa memoria estática no volatil, pues, según su contenido, indican al ALTAIR 535 qué dirección debe ejecutar primero al encender la máquina.

BIBLIOGRAFIA FORTH

Desgraciadamente el número de libros FORTH publicados en España se reducen a solamente dos:

- Lenguaje FORTH para micros (PARANINFO).
- FORTH (INGELEK).

Ambos se publicaron en 1984-5, y solamente pueden obtenerse actualmente en mercados de segunda mano a un precio alrededor de las 400 Ptas.

Si sabéis Inglés a nivel de traducción, entonces estáis de suerte, porque en ese idioma podéis obtener toda la información FORTH que deseéis.
La mejor dirección que os puedo dar es:

FORTH INTEREST GROUP
P.O. BOX 2154
OAKLAND, CALIFORNIA 94621

Por $52 al año vais a recibir una revista bi-mensual llamada FORTH DIMENSIONS, y vais a poder comprar una amplia cantidad de libros y software sobre FORTH de alta calidad con un 10% de Descuento.

Glosario eForth

Derivado de bFORTH by Bill Muench, 1990.

ADVERTENCIA: Información avanzada-- sujeta a cambios.

===

replace

===

Sort order

!"#$%&'()*+,-./digits:;<=>?@Alpha[\]^_`{|}~

===

Attributes Capitalized symbols.

- C la palabra debe ser solamente usada durante la compilación de una definición secundaria.
- D la palabra es una palabra para definir palabras.
- I la palabra es inmediata, y se ejecutara durante la compilación, a menos que una especial acción sea tomada.
- U un valor del usuario.

===

Notas de la pila

(compile \ run \ child ;Return ;Float ; <input stream>)

(before -- after ;R before -- after ;F before -- after ; <string>)

===

Glosario

! (w a --) "store"
Guarda un número de 16-bit en la dirección asignada.

!CSP (--) "set c s p"
Guarda los valores actuales de los punteros de las pilas.

!IO (--) "store i o"
Inicializa el interface serie.

(d -- d) "number sign"
Convierte un digito de un número usando la base actual.
Debe estar dentro de <# y #>.

#> (d -- b u) "number sign greater"
Termina una conversión numérica.

#S (ud -- 0 0) "number sign s"
Convierte todos los digitos de un número usando la base actual.

#TIB (-- a) "number t i b"
Una variable doble del sistema que guarda el tamaño y la dirección
asignada al buffer de entrada del terminal.

$" (-- ; <string> \ -- $) I,C "string quote"
Usada solamente dentro de una definición para compilar una cadena
de caracteres terminada con el carácter "
Durante la ejecución la dirección de la cadena empaquetada es puesta
en la pila.

$"| (-- $) C "string quote primitive"
Retorna la dirección de una cadena compilada.
La ejecución primitiva compilada por "

$," (-- ; <string>) "string comma quote"
Compila una cadena de caracteres dentro del area de código, terminada
por "

$,n ($ --) "string comma n"
Crea un nombre para una definición usando una cadena.
Ajusta el puntero al código a la siguiente celda libre en el area
de códigos, ningún código es compilado.

El nombre no es linkado dentro del diccionario.

$COMPILE ($ --)
Convierte una cadena a una dirección de palabra. Ejecuta la palabra
en modo intérprete o la compila en modo compilar.

$INTERPRET ($ --) "string interpret"
En el nivel intereactivo, si una palabra está definida la ejecuta.
Si no lo está, intenta convertirla en un número, si eso falla, emite
un mensaje de error.

' (-- ca ; <string>) "tick"
Retorna la dirección de código de la palabra que sigue.

'?KEY (-- a) U "tick question key"
Un vector del sistema que indica el estado del dispositivo de entrada.

'BOOT (-- a)
Retorna la dirección de la rutina de arranque del sistema.

'ECHO (-- a) U "tick echo"
Vector al dispositivo eco del sistema.

'EMIT (-- a) U "tick emit"
Vector al dispositivo de salida del sistema.

'EVAL (-- a) "tick eval"
Vector intérprete/compila del sistema.

'EXPECT (-- a) U "tick expect"
Vector a la linea de entrada del sistema.

'NUMBER (-- a) "tick number"

Vector a la conversión de número del sistema.

'PROMPT (-- a) U "tick prompt"
Vector al activo del sistema.

'TAP (-- a) U "tick tap"
Vector a la función opción entrada.

((-- ; <string>) I "paren"
Comienza un comentario. El comentario es terminado por el carácter)
Puede ser usado dentro y fuera de una definición.

* (n n -- n) "star"
Multiplica dos números con signo. Retorna un númeoro de 16-bit con signo.

*/ (n1 n2 n3 -- q) "star slash"
Multiplica n1 por n2 produciendo un número de 32-bit intermedio d.
Luego divide d por n3 produciendo un cociente de 16-bit.

*/MOD (n1 n2 n3 -- r q) "star slash mod"
Multiplica n1 por n2 produciendo un número de 32-bit intermedio d.
Luego divide d por n3 produciendo un resto de 16-bit y un cociente de 16-bit.

+ (w w -- w) "plus"
Suma

+! (n a --) "plus store"
Incrementa el valor de 16-bit en la dirección con n.

, (w --) "comma"
Compila un valor de 16-bit dentro del area de códigos.

- (w w -- w) "minus"
Resta el valor que hay arriba de la pila al segundo de la pila.

-TRAILING (b u -- b u) "dash trailing"
Ajusta el contador para eliminar cualquier carácter en blanco al comienzo de una cadena.

. (n --) "dot"
Muestra un valor de 16-bit usando la actual base. Si BASE es DECIMAL, muestra un número con signo.

." (-- ; <string>) I,C "dot quote"
Usado solamente dentro de una definición para compilar una cadena empaquetada terminada por el carácter "
En modo ejecución, la cadena es mostrada en el dispositivo actual de salida de datos.

."| (--) C "dot quote primitive"
Muestra una cadena compilada en el dispositivo de salida. Esta primitiva de ejecución es compilada por ."

.((-- ; <string>) I "dot paren"
Comienza un comentario que es mostrado en el dispositivo actual.
Puede ser usado dentro y fuera de una definición.

.ID (na --) "dot i d"
Muestra la cadena empaquetada en la dirección na.

.OK (--) "dot o k"
Muestra el mensaje ok

.R (n +n --) "dot r"
Muestra un valor de 16-bit n en un campo de un ancho +n justificado

```
        a la derecha, usando la base actual.

.S      ( ? -- ? )      "dot s"
        Muestra el contenido de la pila.

/       ( n n -- q )    "slash"
        Divide dos números de 16-bit. Retorna solamente un cociente de 16-bit.

/MOD    ( n n -- r q )  "slash mod"
        Divide dos números de 16-bit. Retorna un resto de 16-bit y un cociente
        de 16-bit.


0<      (n -- t )       "zero less"
        Retorna verdadero si w es menor que 0, negativo. La comparación
        es con signo. También es usado para extensión de signo.

0=      ( w -- t )      "zero equals"
        Retorna verdadero si w es igual a 0.

2!      ( d a -- )      "two store"
        Guarda un número de 32-bit en una dirección.

2@      ( a -- d )      "two fetch"
        Retorna un número de 32-bit almacenado en una dirección.

2DROP   ( d -- )        "two drop"
        Saca un número de 32-bit de la pila, o que es lo mismo, dos números
        de 16-bit.


2DUP    ( d -- d d )    "two dupe"
        Duplica un número de 32-bit en lo alto de la pila.

:       ( -- ; <string> )       D       "colon"
        Comienza la definición de una palabra secundaria para ser anexionada
        al vocabulario actual.

;       ( -- )  I,C     "semicolon"
        Termina la definición de una palabra que comienza con :

<       ( n1 n2 -- t )  "less than"
        Retorna verdadero si n1 es menor que n2. La comparación es con signo.

<#      ( -- )  "start number"  "less number sign"
        Comienza una conversión numérica.

=       ( w w -- t )    "equals"
        Retorna verdadero si w1 es igual a w2.

>CHAR   ( u -- c )      "to character"
        Convierte un valor en un carácter imprimible. Cambia un carácter
        no imprimible por el carácter _

>IN     ( -- a )        "to in"
        El puntero dentro de la  cadena de entrada actual.

>NAME   ( ca -- na, F ) "to name"
        Si es posible, convierte una dirección de código a una dirección de
        nombre. Si no es posible, retorna falso.

>R      ( w -- ;R -- w )        C       "to r"
        Pone el valor de 16-bit que hay en lo alto de la pila en la pila de
        retorno.

?       ( a -- )        "question"
        Muestra un valor de 16-bit almacenado en una dirección, usando la
        base actual. Si BASE es DECIMAL, es mostrado con signo.

?branch ( f -- )        "question branch"
```

Una rutina que en modo ejecución redirecciona la ejecución a la dirección de la siguiente celda si la banderín es verdadero.

?CSP (--) "question c s p"
Compara el valor actual de los punteros de las pilas con los valores guardados. ABORT con un mensaje de error si son diferentes.

?DUP (w -- w w, 0) "question dupe"
Duplica el número que hay en lo alto de la pila solamente si no es cero.

?KEY (-- t) "question key"
Retorna el estado del dispositivo de entrada.

?RX (-- c T, F) "question r x"
Retorna un carácter del dispositivo de entrada y verdadero.
Retorna falso si no hay carácter pendiente.

?STACK (--) "question stack"
Muestra un mensaje de error si los limites de la pila han sido pasados.

?UNIQUE ($ -- $) "question unique"
Muestra un mensaje de advertencia cuando hay una definición duplicada.

@ (a -- w) "fetch"
Retorna un valor de 16-bit guardado en una dirección.

@EXECUTE (a --) "fetch execute"
Busca el token de ejecución guardado en la dirección y lo ejecuta. (ejecución indirecta). Si el valor contenido es cero, no hace nada.

ABORT (--)
Reajusta la pila y ejecuta QUIT (Ningún mensaje es mostrado).

ABORT" (-- ; <string> \ f --) I,C "abort quote"
Usada solamente dentro de la definición para compilar una cadena empaquetada terminada con el carácter "
En modo ejecución, si el banderín es falso, ejecuta la secuencia de palabras que sigue a la cadena. Si el banderín es verdadero, la cadena es mostrada en el dispositivo actual de salida, y la ejecución es pasada a una rutina que maneja errores.

abort" (f --) C "abort quote primitive"
La primitiva de ejecución compilada por ABORT"

ABS (n -- +n)
Retorna el valor absoluto de n.

accept (b u -- b u)
Recibe una linea de un máximo de u caracteres a una entrada de buffer en una dirección byte. Termina la entrada si un CR es recibido. Retorna el actual contador de carácteres recibidos.
Usa el actual dispositivo de entrada.

AFT (a -- a a \ --) I,C
Usado dentro de una estructura de bucle para saltar un trozo de código en el primer tiempo del bucle. AFT compila la instrucción de salto incondicional y deja la dirección para ser resuelta por THEN.

AGAIN (a -- \ --) I,C
Termina una estructura de salto infinito. AGAIN compila una instrucción de salto incondicional, y usa la dirección izquierda a BEGIN para resolver este salto hacia atrás.

AHEAD (-- a \ --) I,C
Marca el comienzo de un salto hacia delante, estructura de salto

incondicional. AHEAD compila la instrucción de salto incondicional
y deja una dirección para ser resuelta por THEN.

ALIGNED (b -- a)
Convierte una dirección byte en una dirección palabra alineada.

ALLOT (n --)
Ajusta el puntero de area de código por n.

AND (w w -- w)
Un AND lógico.

BASE (-- a) U
La variable del sistema que guarda la base numérica para la conversión.

BEGIN (-- a \ --) I,C
Marca el comienzo de una estructura bucle indefinida. Deja una dirección para ser resuelta por UNTIL, WHILE y REPEAT.

BL (-- c) "b l"
Deja el valor de un espacio en la pila, el carácter ASCII 32.

branch (--) C
La rutina de ejecución para de redirigir la dirección de ejecución de la próxima celda.

BYE (--)
Termina FORTH y retorna al sistema operativo.

C! (v b --) "c store"
Guarda un valor de tamaño byte en una dirección.

C@ (b -- v) "c fetch"
Retorna el valor tamaño byte guardado en la dirección.

CALL, (a --) C "call comma"
Ensambla una rutina de 4 bytes que llama a una dirección señalada.

CATCH (ca -- err#/0)
Ajusta un valor de error local y ejecuta la palabra referenciada por la ejecución del token ca. Retorna número de error o un cero si no hay error.

CELL+ (a1 -- a2) "cell plus"
Suma el tamaño de una celda en bytes a la dirección a1.

CELL- (a1 -- a2) "cell minus"
Resta el tamaño de una celda en bytes a la dirección a1.

CELLS (n1 -- n2)
Multplica n1 por el tamaño de una celda en bytes.

CHAR (-- c ; <string>)
Retorna el valor del primer carácter en <string>. Si es usado dentro de una definición, usa la frase [CHAR <tring>] LITERAL

CHARS (+n c --)
Muestra el carácter +n en el dispositivo salida actual.

CMOVE (b1 b2 u --) "c move"
Mueve u valores tipo byte desde la dirección b1 a b2, procesando desde más bajo a mayor memoria. Sobreescribe ocurre si b1<b2<b1+u

COLD (--)
Reinicia completamente el sistema, pero no recarga el sistema desde la ROM.

COMPILE (--) C

Usada solamente dentro de una definición. En ejecución la palabra que sigue a COMPILE no es ejecutada, pero su dirección de código es copiada dentro del area de códigos.

CONSOLE (--)
Inicializa los vectores de entrada y salida a un terminal.

CONTEXT (-- a)
La variable usada para especificar el diccionario para buscar.

COUNT ($ -- b +n)
Retorna la dirección y un contador tamaño byte de la cadena empaquetada.

CP (-- a) "c p"
El puntero a la próxima localización disponible en el espacio para códigos. Desde que códigos y nombre están separados, el tradicional DP, puntero diccionario, ha sido separado en CP, puntero código, y NP, puntero nombre.

CR (--) "carriage return"
La posición del cursor al comienzo de la nueva linea en el dispositivo de salida.

CREATE (-- ; <string> \ -- a) D
Construye una definición. En ejecución, la dirección apuntando al siguiente espacio de código es dejada en la pila.

CSP (-- a) "c s p"
La variable del sistema que guarda el puntero actual de la pila. Usada para comprobar errores.

CURRENT (-- a)
La variable usada para especificar el vocabulario en el cual las nuevas definiciones van a ser compiladas.

DECIMAL (--)
Ajusta BASE a decimal (base 10).

DEPTH (-- n)
El número de elementos en la pila, no incluye n.

DIGIT (u -- c)
Convierte a un único dígito (número) a su valor ASCII.

DIGIT? (c base -- v t) "digit question"
Intenta convertir un carácter a dígito binario. Retorna verdadero si el dígito es valido para la base corriente.

dm+ (a u -- p a+) "d m plus"
Muestra los valores de 16-bit comenzando en la dirección alineada a.

DNEGATE (d -- -d) "d negate"
Retorna el complemento a dos de un número de 32-bit. Cambia el signo de un número de 32-bit.

do$ (-- $) C "do string"
Retorna la dirección de una cadena empaquetada compilada.

doLIST (a --) C "do list"
La rutina de ejecución que ejecuta la lista definida apuntada por a.

doLIT (-- n) C "do literal"
Retorna el literal compilado por LITERAL.

doUSER (--) C "do user variable"
La acción ejecución de las variables usuario.

doVAR (-- a) C "do variable"

```
         La acción ejecución de una variable.

DROP     ( w -- )
         Quita el elemento que hay en lo alto de la pila.

DUMP     ( a u -- )
         Muestra valores HEX y carácter comenzando en la dirección alineada a,
         para el contador u.

DUP      ( w -- w w )    "dupe"
         Duplica el valor que hay en lo alto de la pila.

ELSE     ( -- \ a -- a ) I,C
         Usada dentro de una estructura de salto incondicional. ELSE resuelve
         el salto condicional hacia delante por IF. ELSE luego compila una
         instrucción de salto incondicional, dejando una dirección para ser
         resuelta por THEN.

EMIT     ( w -- )
         Salida de un carácter en el actual dispositivo de salida.

EVAL     ( -- )
         Intérpreta o compila tokens desde el canal de entrada.

EXECUTE ( w -- ;R -- w )
         Ejecuta una palabra definida indicada por la ejecución del token w.

EXIT     ( -- ;R w -- )
         Compila una subrutina retorno.

EXPECT   ( b u -- )
         Recibe una linea de un máximo de u caracteres  en la dirección b
         del buffer de entrada. Termina la entrada si un retorno carro es
         recibido. El contador de caracteres recibidos es salvado en la variable
         SPAN. Usa el dispositivo entrada actual. Esta palabra es vectorada.

EXTRACT ( d base -- d' c )
         Usada incrementalmente para convertir cada dígito de un número a su
         valor carácter.

FILE     ( -- )
         Especifica la entrada del sistema desde un fichero usando "pace
         handshake". La entrada desde fichero no tiene eco, todos los
         mensajes son mostrados.

FILL     ( b u v -- )
         LLena un area en la dirección b de longitud u usando el valor v.

find     ( $ va -- ca f, $ F )   "find primitive"
         Dada una cadena y un diccionario entrada entrelazado, busca un
         nombre igual a la cadena. Si encontrado, retorna la dirección de código
         y una bandera verdadera. Si no encontrado, la dirección de la cadena y
         una bandera falsa.

FOR      ( u -- )       I,C
         Comienza un bucle con un contador hacia abajo. Repite el bucle hasta
         NEXT u+1 veces desde u a 0.

HAND     ( -- )
         Especifica la entrada del sistema desde el teclado, no handshake.
         Todas las entradas tienen eco.

HANDLER ( -- a )
         El actual puntero manejador de errores.

HERE     ( -- a )
         Pone en la pila la siguiente dirección libre en el area de códigos.

HEX      ( -- )
         Ajusta BASE a hexadecimal (Base 16):
```

```
hi       ( -- )
         Muestra el mensaje copyright.

HLD      ( -- a )        "h l d"
         El puntero a la salida de una cadena numérica formateada.

HOLD     ( c -- )
         Inserta el carácter en la formateada cadena numérica.

I/O      ( -- a )        "i slash o"
         Una matriz usada por CONSOLE para inicializar los vectores de
         entrada y salida del sistema. El vector pedido es 'KEY?
         'KEY y 'EMIT.

IF       ( -- a \ f -- ) I,C
         Marca el comienzo de un salto hacia delante, estructura condicional.
         IF compila la instrucción de salto condicional y deja una dirección
         para ser resuelta por THEN o ELSE.

IMMEDIATE       ( -- )
         Marca la más reciente entrada hecha en el diccionario como una
         palabra que será ejecutada durante la compilación.

INVERT   ( w -- w )
         Invierte los bits. Equivalente a -1 XOR. El complemento a uno.

KEY      ( -- c )
         Retorna un carácter desde el actual dispositivo de entrada.

kTAP     ( b b b c -- b b b )    "k tap"
         La rutina 'tap usada por la entrada desde fichero.

LAST     ( -- a )
         El puntero al nombre de la entrada  más reciente en el diccionario.

LITERAL ( w -- \ -- w ) I
         Compila un número como un valor en linea.

M*       ( n n -- d )    "m star"
         Multiplica dos número con signo. Retorna un número de 32-bit.

M/MOD    ( d n -- r q )  "m slash mod"
         Una división de un número de 32-bit dividido por un número de 16-bit.
         Retorna un cociente de 16-bit un resto de 16-bit.

MAX      ( n n -- n )
         Deja el más grande de los dos números con signo.

MIN      ( n n -- n )
         Deja el más pequeño de los dos números.

MOD      ( n n -- r )
         División de dos números de 16-bit. Retorna solamente un resto de 16-bit.

NAME>    ( na -- ca )    "name to code"
         Convierte una dirección de nombre a una dirección de código.

NAME?    ( $ -- ca f, $ F )      "name question"
         Dada una cadena, busca un nombre igual.  Si encontrado, retorna
         la dirección de código y la bandera verdadera. Si no encontrada,
         la dirección de la cadena y una bandera falsa.

NEGATE   ( n -- -n )
         Equivalente a 0 SWAP -
         Es el complemento a dos de un número. Cambia el signo de un número.

NEXT     ( a -- \ -- ;R u -- [u-1] )     I,C
         Termina una estructura bucle con contador hacia abajo. NEXT compila
         la instrucción de salto a la dirección que apunta a la izquierda de FOR.
```

El contador es guardado en la pila de retorno. El bucle continua hasta que el contador sea igual a cero.

```
next    ( -- )  C
        Rutina ejecucicón para terminar un contador hacia abajo de un bucle
        FOR.....NEXT. Ver NEXT.

NP      ( -- a )        "n p"
        El puntero a la siguiente localización disponible en el espacio
        de nombres en el diccionario.

NUF?    ( -- t )        "nuf question"
        Continua hasta pausa o fin por el usuario. Cualquier tecla será
        pausa, una vez en pausa cualquier tecla excepto ENTER será recomenzar
        y retorna falso, ENTER será verdadero.

NULL$   ( -- $ )        "null string"
        La dirección de una cadena con un contador cero.

NUMBER? ( $ -- d T, $ F )       "number question"
        Intenta convertir una cadena empaquetada a un número binario. Si
        es posible retorna el número y verdadero. En caso contrario, retorna
        la dirección de la cadena y falso

OR      ( w w -- w )
        Un lógico OR a nivel del bits.

OVER    ( w1 w2 -- w1 w2 w1 )
        Copia el segundo elemento en la pila en lo alto de la pila.

OVERT   ( -- )
        Usada por ; para unir una palabra definida con éxito.

PACE    ( -- )
        Envia el carácter handshake fichero transferencia.

PACK$   ( b u $ -- $ )  "pack string"
        Mueve y convierte la cadena en la dirección b con el contador u en una
        cadena empaquetada en la dirección $.

PAD     ( -- a )
        Dirección de un buffer temporal.

PARSE   ( c -- b u ; <string> )
        Escanea el canal de entrada actual para un dado carácter como
        delimitador. Retorna la dirección de comienzo b y el contado u
        de una cadena delimitada.

parse   ( b u c -- b u delta ; <string> )
        Escanea una cadena para un dado carácter como delimitador. Retorna el
        comienzo de una dirección b y un contador u de la cadena delimitada.
        Delta es el comienzo del actual offset.

PICK    ( +n -- w )
        Copia el valor +nth en la pila en lo alto de la pila.

PRESET  ( -- )  C
        Limpia la pila, la pila de  retorno e inicializa el sistema.

QUERY   ( -- )
        Recibe el actual buffer de entrada una linea desde el actual
        dispositivo de entrada.

QUIT    ( -- )
        Limpia la pila de retorno, ajusta el estado del intérprete, y retorna
        el control al intérprete de comandos.

R>      ( -- w ;R w -- )        C       "r from"
        Coge el elemento que hay en lo alto de la pila y lo pone en la pila de
        retorno.
```

R@ (-- w ;R w -- w) "r fetch"
Copia el elemento que hay en lo alto de la pila de retorno y lo deja en la pila.

RECURSE (-- \ -- ;R -- a) I,C
Usada solamente dentro de una palabra siendo definida para permitir auto-llamarse. Recursión.

REPEAT (a a -- \ --) I,C
Termina una estructura de salto indefinida. REPEAT compila una instrucción, y usa la dirección izquierda a WHILE para resolver este salto hacia atrás.

ROT (w1 w2 w3 -- w2 w3 w1) "rote"
Rota los tres elementos que hay en lo alto de la pila. El tercer elemento va a lo alto y los otros van hacia abajo.

RP! (a --) C "r p store"
Ajusta el puntero de la pila de retorno a la dirección.

RP0 (-- a) "r p zero"
Retorna la dirección del fondo de la pila de retorno.

RP@ (-- a) "r p fetch"
Retorna la dirección del puntero de la pila de retorno.

SAME? (a1 a2 u -- a1' f \ -0+) "same question"
Compara las dos cadenas, retorna la dirección de comienzo de la primera cadena y una bandera verdadera.

SIGN (n --)
Muestra el signo menos si n es negativo. Deberia ser usado dentro de <# y #>.

SP! (a --) "s p store"
Ajusta el puntero de la pila a la dirección a.

SP0 (-- a) "s p zero"
Retorna la dirección del fondo del puntero de la pila.

SP@ (-- a) "s p fetch"
Retrona la dirección del puntero de la pila.

SPACE (--)
Muestra un espacio, el carácter ' ' en el actual dispositivo de salida.

SPACES (+n --)
Muestra +n espacios en el actual dispositivo de salida.

SPAN (-- a)
La variable del sistema que guarda la cantidad de caracteres introducidos por EXPECT.

str (d -- b u) "s t r"???
Convierte un número a una cadena usando la base actual. Con si la base es decimal.

SWAP (w1 w2 -- w2 w1)
Intercambia los dos elementos que hay en lo alto de la pila.

TAP (b b b c -- b b b')
Eco y guarda la tecla-código pulsada, y actualiza la posición del cursor.

temp (-- a) U
Retorna la dirección de una variable de usuario para almacenamiento temporal.

THEN (a -- \ --) I,C
Termina una estructura de salto condicional. Resuelve un salto hacia delante compilado por IF, ELSE, AHEAD o AFT.

THROW (err# -- err#)
Reajusta el estado del sistema al actual error, y actualiza la bandera error.

TIB (-- a) "t i b"
Dirección del buffer de entrada del terminal.

TOKEN (-- $; <string>)
Escanea el canal de entrada actual para una palabra delimitada por un carácter en blanco. Mueve la palabra al final del fin del area de nombres como una cadena empaquetada. Retorna la dirección de la cadena empaquetada.

TX! (c --) "t x store"
Envia un carácter al dispositivo de salida. Primitiva de EMIT.

TYPE (b u --)
Salida de u caracteres de la cadena en la dirección b por el actual dispositivo de salida.

U. (u --) "u dot"
Muestra un número de 16-bit sin signo, usando la base actual.

U.R (u +n --) "u dot r"
Muestra un número de 16-bit sin signo justificado a la derecha en un campo de un ancho de +n, usando la actual base.

U< (u1 u2 -- t) "u less"
Retorna verdadero si u1 es menor que u2. La comparación es sin signo.

UM* (u u -- ud) "u m star"
Mulplica dos números sin signo de 16-bit. Retorna una número de 32-bit.

UM+ (u u -- ud) "u m plus"
Suma dos números sin signo y retorna una suma de 32-bit.

UM/MOD (ud u -- ur uq) "u m slash mod"
División sin signo de un número de 32-bit dividido por un número de 16-bit. Retorna un cociente sin signo de 16-bit y un resto de 16-bit sin signo.

UNTIL (a -- \ f --) I,C
Termina una estructura bucle indefinida. Una condición es comprobada después de ejecutar el código dentro del bucle. UNTIL compila una instrucción de salto condicional, y usa la dirección a la izquierda de BEGIN para resolver este salto hacia atrás.

UP (-- a) "u p"
Retorna la dirección de la actual area de usuario.

USER (u -- ; <string> \ -- a) D
Construye una variable usuario con un offset desde la actual base usuario. En ejecución la dirección de la variable es dejada en la pila.

VARIABLE (-- ; <string> \ -- a) D
Construye una variable. En ejecución, la dirección de la variable es dejada en la pila.

VER (-- n)
Retorna el código versión. La mayor revisión es en el byte alto y la menor realización es en el byte bajo.

WHILE (a -- a a \ f --) I,C

Usada dentro de una estructura bucle indefinido. Una condición es comprobada antes de ejecutar el código dentro del bucle. WHILE compila la instrucción de salto condicional y deja una dirección para ser resuelta por REPEAT.

WITHIN (u lo hi -- t)
Retorna verdadero si lo <= u < hi. Comparación sin signo y circular.

WORD (c -- $; <string>)
Escanea el actual canal de entrada para la cadena delimitada por 'c'. Retorna la dirección de la cadena empaquetada.

WORDS (--)
Muestra las palabras que hay en el diccionario CONTEXT. Continua mostrando hasta que termina, o una tecla pausa o terminación es pulsada por el usuario.

XIO (a1 a2 a3 --) "x i o"
Revector 'prompt', 'echo' y 'tap' a las direcciones de código en la pila.

XOR (w w -- w)
OR lógico exclusivo a nivel de bits.

[(--) I "left bracket"
Comienza interprentado el texto desde el canal de entrada. Cambia de compilando a interpretando.

[COMPILE] (-- ; <string> \ --) I "bracket compile"
Usado solamente dentro de una definición para forzar la compilación de una palabra IMMEDIATE.

\ (-- ; <string>) I "backslash"
Comienza un comentario. El comentario es terminado por el fin de linea del sistema en el final de una linea. Puede ser usado dentro y fuera de una compilación.

] (--) "right bracket"
Cambia de interpretando a compilando.

^H (b b b -- b b b' ; <backspace>) "control h"
Una macro para el teclado que borra caracteres desde el actual canal de entrada. No es tomada ninguna acción si el comienzo del canal de entrada es alcanzado.

_TYPE (b u --) "printable type"
Muestra el comienzo de una cadena en la dirección b, con contador u. Sustituye con _ cualquier carácter no imprimible.

OFERTAS

DISCOS QL DE DOMINIO PUBLICO

Descripción
===========

CUQ_A1	Números	1-2-3-4
CUQ_A2	"	5-6-7-8
CUQ_A3	"	9-10
CUQ_A4	"	11-12
CUQ_A5	"	13-14
CUQ_A6	"	15-16
CUQ_A7	"	17 + Screens

```
CUQ_A8     "    18-19
CUQ_A9     "    20-21
CUQ_A10    "    22-23
CUQ_A11    "    24-25
CUQ_A12    "    26-27
CUQ_A13    "    28-29
CUQ_A14    "    30-31
CUQ_A15    "    32
CUQ_A16    "    33
CUQ_A17    "    35
SCREEN_1    Pantallas digitalizadas
SCREEN_2          "      "
SCREEN_3          "      "
SCREEN_4          "      "
SCREEN_5    Pantallas Spectrum
GRAFICOS PARA ADULTOS 1, 2
COLECCION GRAFICOS COMPRIMIDOS
TRAINING V5.1
QLIPER_A1 números 36-38
QLIPER_A2    "    37-40
QLIPER_A3    "    39
QLIPER_A4    "    41
QLIPER_A5    "    42
QLIPER_A6    "    43
QLIPER_A7    "    44
QLIPER_A8    "    45
QLIPER_A9    "    46
QLIPER_A10   "    47


QLAVE
=====

QLAVE   1-2-3


QUANTA
======
C.A.D_1
COMMS_XFER_1
COMMS_XFER_2
EDUC_1
GMS_STRAT_4
GRAPHICS_1
GRAPHICS_2
KERMIT      1-2-3
LANGUAGES_1
MATHS_1
QDOS_JS_1     (QDOS JS ROM DISASSEMBLY)
UTIL_DRCPY
UTIL_EMACS_1   (UTILs Micro-EMACS editor)
UTIL_EMACS_2   (Run version editor)
UTIL_GEN 1, 2, 3, 4
PSION 1, 2, 3, 4
VT 1, 2
PF 1, 2, 3
SP 1

C.G.H. Services
===============

C001-C002-C004-C005-C006-C007
AUSTRALIAN 1
CONECTIONS 1
COMMS 1
IMAGENES GIF 1, 2, 3
SUPERBASIC UTILITIES 1
PROG_LANGUAGE 1

New England QL User Group (NESQLUG)
```

===================================

A001

Svenska QL Gruppen (SveQL)
==========================

S001-S002-S003-S004-S005-S006

QL Contact France
=================

F001-F002-F003-F004-F005-F006-F007-F008-F009-F010-F011-F012-F013

QITALY CLUB
===========

I001-I002-I003-I004-I005-I006-I007-I008-I009-I010-I011-I012-I013
I014-I015

Qitaly Magazine 24-25-26-27

QUBBE
=====

Q001-Q002
MOLECULAR GRAPHIC v2.0
ELVIS EDITOR
Text 'N' Graphix (Demo)
QPACer
ZM1+ Spectrum Emu

Scottish QL Users Group (SQLUG)
===============================

T001

National Dutch QL-Users Club (sin_QL_air)
===

H001-H002-H003-H004-H005-H006-H007

QUASAR
======

QUASAR 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12

Individual Software (Diferentes origenes)
===

X001-X002-X003-X004-X005
QBOX v1.19b
SPECTATOR v1.35 (Spectrum 128K)
Software Spectrum discos 1, 2, 3
Xtricator v1.10
software ZX81 discos 1, 2
PSION XCHANGE v3.90
GZIP
SPECULATOR (48K)
YACC
QTPI

Compilador C68 v4.12 en versión runtime (3 discos).
Compilador C68 para QDOS (v4.01 completa, y runtime v3.05).- Son 5 discos con el código Fuente. Solamente son necesarios 3 que contienen el runtime, varias utilidades y los manuales.

LIB CPORT
LIB QPTR
C Debugging Tools

C Programming Tools 1
LIB CURSES
GNU Text Utilities 1, 2

Condiciones:

- Debe enviar 100 ptas por cada disco a copiar. Esto incluye copia y embalaje/sellos (200 Ptas para Europa y 250 Ptas Resto Mundo).
- Naturalmente todos los discos enviados deben estar formateados a 1440 sectores. En otras palabras, no deben tener sectores defectuosos.
- La forma de pago para estas pequeñas cantidades es preferible en monedas o billetes.
- Para aquellos que no deseen enviarme los discos, puedo ofrecerlos grabados por 200 ptas (300 Ptas a Europa, y 350 Ptas Resto Mundo).
- Existe un fichero conteniendo una lista de todos los programas disponibles en los diferentes discos de nuestra libreria.

```
+------------+
| INVENTARIO |
+------------+
```

==
CAJA QLIPER
==

Fecha	Concepto	ESP	Saldo
93.01.01	Saldo anterior		+12355
93.12.21	Franqueo QLíper 46	-1386	+10969
93.12.28	Envio material sobrante a S. Merino	-2207	+8762
94.01.04	Derechos correos y telegrafos	-204	+8558
93.12.30	Franqueo QLíper 47	-1505	+7053
94.01.05	30 discos	-1980	+5073
94.01.05	Pedido atrasados de Francisco Diaz	+4000	+9073
94.01.05	Francisco Diaz-Tendero	+1500	+10573
94.01.05	Pablo Cardenes Cañeque	+1500	+12073
94.01.07	Franqueo pedido Francisco Diaz	-410	+11663
94.01.11	20 Etiquetas 50x75 KORES Fixo	-100	+11563
94.01.11	Dasio Carballeira Tella	+2000	+13563
94.01.11	Javier Zubieta Aguirre	+1500	+15063
94.01.12	20 discos	-1320	+13743
94.01.12	Franqueo	-255	+13488
94.01.13	Franqueo	-135	+13353
94.01.14	Sellos	-90	+12263
94.01.14	Marcos Cruz Martín	+1500	+14763
94.01.14	Miguel Estarellas	+1500	+16263
94.01.14	Joaquín Gallardo Rodríguez	+1500	+17763
94.01.17	Pedro Reina	+2000	+19763
94.01.18	Josu Regidor Eguren	+1500	+21263
94.01.18	Sobres	-30	+21233
94.01.18	Franqueo	-45	+21188
94.01.20	Sobres	-145	+21043
94.01.20	Franqueo	-585	+20458
94.01.20	Ian-Charles Coleman	+3200	+23685
94.01.21	Franqueo	-345	+23340
94.01.25	Franqueo intercambio y pedido Ian	-425	+22915
94.01.25	Eduard Losada	+2000	+24915
94.01.26	Franqueo 49	-705	+24210
94.01.26	20 discos	-1320	+22890
94.01.31	5 sobres	-25	+22865
94.01.31	Publicidad ex-socios'91	-308	+22557
94.01.31	Franqueo	-110	+22447
94.01.31	Rafael Illanes Muñoz	+1500	+23947

94.01.31	20 discos	-1320	+22627
94.02.01	Felipe Berganza Picón	+2000	+24627
94.02.01	Franqueo	-65	+24562
94.02.01	17 sobres	-85	+24477
94.02.01	40 etiquetas Kores Fixo 50x75	-200	+24277
94.02.02	Franqueo 49 a Sinclair QL World	-135	+24142
94.02.02	40 etiquetas Kores Fixo 50x75	-200	+23942
94.02.03	Pedido de Joaquin	+1000	+24942
94.02.04	Franqueo Joaquin	-105	+24837
94.02.04	Publicidad ex-socios'93	-180	+24657
94.02.10	Franqueo 49 a Clubes	-740	+23917
94.02.14	Felix Alonso	+2500	+26417
94.02.14	Franqueo	-65	+26352
94.02.14	20 discos	-1320	+25032
94.02.15	Envio QLíper 49 a Qubbesoft	-135	+24897
94.02.23	Franqueo (Retorno material y un pedido)	-235	+24662
94.03.04	Franqueo QLíper 50	-1465	+23197
94.03.04	19 sobres	-114	+23083
94.06.14	9 sobres	-54	+23029
94.06.14	Franqueo QLíper 51	-1060	+21969
94.08.01	Franqueo Qitaly	-135	+21834
94.09.29	Envio Juanjo	-46	+21788
94.10.15	20 sobres	-100	+21688
94.10.15	20 discos HD	-1320	+20368
94.10.15	Sellos	-1143	+19225

==

MATERIAL QLIPER

==

Fecha	Sobres	Sobres-Acolchados	Discos	DISCOS-ATRASADOS-PD-SHARE
94.01.07	6	7	18	172
94.01.11	2	6	14	172
94.01.12	2	6	34	172
94.01.14	-2	5	30	172
94.01.18	2	5	29	174
94.01.20	25	5	25	174
94.01.21	23	5	23	175
94.01.25	22	5	22	177
94.01.26	8	5	31	177
94.01.31	4	5	49	177
94.02.01	20	5	47	177
94.02.02	20	4	46	177
94.02.03	20	4	36	187
94.02.04	16	3	32	187
94.02.10	11	3	27	187
94.02.14	10	3	45	187
94.02.15	9	3	44	187
94.02.23	8	2	39	192
94.02.04	8	2	20	192
94.06.14	0	2	3	192
94.08.01	-1	2	2	193
94.09.23	-2	2	1	193
94.10.15	1	2	3	194